基于差分进化的多目标粒子群特征选择算法 *

李　敏 [a, b]，章国豪 [a]，陈梓樑 [b]，郭志勇 [b]，胡晓敏 [b†]

(广东工业大学 a. 信息工程学院; b. 计算机学院，广州 510006)

摘　要：特征选择技术在大数据分析、图像处理、生物信息学等领域具有重要作用。在实际应用中，降低分类错误率和减少提取出的特征数量便于后续数据的利用，往往是两个冲突的目标。基于拥挤、变异和支配策略的多目标粒子群特征选择（crowding，mutation，dominance particle swarm optimization for feature selection，CMDPSOFS）算法是一种面向特征选择应用中特征数量最小和分类错误率最低的双目标优化算法。它使用三种不同的变异机制，用于保持群体多样性和平衡全局、局部搜索的能力，但其中的均匀变异使算法的随机性大大增加，产生较多适应值差的解，降低了算法收敛速度。改进的 CMDPSOFS-II 算法将差分进化算法中的变异算子和选择操作引入到 CMDPSOFS 算法中，实验结果表明 CMDPSOFS-II 算法在特征选择上得到比原来的方法更优的结果，更好地平衡了全局和局部搜索能力。

关键词：特征选择；粒子群算法；变异；差分进化

中图分类号：TP301.6　　doi: 10.19734/j.issn.1001-3695.2018.05.0448

Multi-objective particle swarm optimization algorithm using differential evolution for feature selection

Li Min[a, b], Zhang Guohao[a], Chen Ziliang[b], Guo Zhiyong[b], Hu Xiaomin[b†]

(*a. School of Information Engineering, b. School of Computers Guangdong University of Technology, Guangzhou* 510006, *China*)

Abstract: Feature selection technology plays an important role in big data analysis, image processing, bioinformatics and other fields. In practical applications, the objectives of reducing the classification error rate and reducing the number of extracted features for facilitating the use of subsequent data, are often two conflicting goals. The multi-object particle swarm optimization based on crowding, mutation, dominance for feature selection (CMDPSOFS) is a kind of bi-objective optimization algorithm with the minimal number of features and classification error rate in feature-oriented selection applications. The algorithm uses three different mutation mechanisms for maintaining swarm diversity and balancing global and local search capabilities. However, the uniform variation increases the randomness of the algorithm, resulting in the generation of worse solutions, which reduces the convergence speed of the algorithm. This paper proposed an improved CMDPSOFS-II algorithm to introduce the mutation and selection operations of differential evolution algorithm into the CMDPSOFS algorithm. The experimental results show that the CMDPSOFS-II algorithm is superior to the original method in feature selection and better balances global and local search capabilities.

Key words: feature selection; particle swarm optimization; mutation; differential evolution

0　引言

分类，作为在机器学习和数据挖掘中的重要步骤，其作用是根据数据集中的特征来将每个实例分类到不同的集合中[1]。一般来说，在没有任何先验知识的情况下，人们很难去确定数据集中哪些特征是对分类有效的。一个数据集经常会引入大量的特征，这里面包含很多有关的、无关的和多余的特征。对于分类来说，无关的和多余的特征显然没有意义，甚至会因为搜索空间的增大而导致分类性能下降。通过特征选择，提取数据集中有代表性的特征，是缩短分类器的训练时间和提高分类能力的常用手段[2]。

特征选择技术在大数据分析[3]、图像处理[4]、生物信息学[5]等领域具有重要作用。在一个数据集中，随着特征数的增多，搜索空间的大小呈指数增长，大多数情况下不可能做到穷

收稿日期：2018-05-15；**修回日期：**2018-06-29　**基金项目：**国家自然科学基金资助项目（61772142，61574049）；广州市珠江科技新星项目（201806010059）

作者简介：李敏（1978-），女，湖北荆州人，博士研究生，主要研究方向为智能计算、图像处理；章国豪（1964-），男，教授，博士，主要研究方向为信号处理、芯片设计；陈梓樑（1996-），男，本科，主要研究方向为智能计算、数据挖掘；郭志勇（1995-），男，本科，主要研究方向为智能计算、数据挖掘；胡晓敏（1983-），女（通信作者），副教授，博士，主要研究方向为智能计算、数据挖掘（xmhu@ieee.org）.

举搜索。为了解决这个问题，许多搜索方法被应用到特征选择中。如 Wang 等人[6]提出应用在特征选择的新型细菌算法，Belciug 等人[7]提出的基于回归的特征选择算法，段洁等人[8]提出了基于邻域粗糙集的多标记分类特征选择算法和谢娟英等[9]提出的基于 DFS 与 SVM 的特征选择算法。

粒子群优化（particle swarm optimization，PSO）作为鲁棒性和适用性高的群体智能优化方法，也被应用到特征选择中。Naeini 等人[4]利用粒子群算法用于高空间分辨率卫星图像分类，选择出的特征组合获得了更高的识别率。Zhang 等人[10]提出一种基于群体智能的算法用于解决声学缺陷检测中的特征选择问题。在实际应用中，除了降低分类错误率这单一目标之外，往往还需要减少提取的特征数量，降低获得所需特征取值的总成本。这些特征需求与降低分类错误率，往往是两个冲突的目标。

多目标优化是求解多个冲突优化目标的有效手段。Zhang 等人[11]提出多目标粒子群算法解决基于代价的特征选择问题，用于提高分类能力并且最小化特征涉及的代价这两个目标。Xue 等人[12]把最小化提取的特征数量和分类错误率作为特征选择问题的两个目标，并提出两种多目标粒子群算法，分别是基于非支配排序的特征选择粒子群算法（nondominated sorting PSO for feature selection， NSPSOFS）和基于拥挤、变异和支配的特征选择粒子群算法（crowding, mutation, and dominance PSO for feature selection，CMDPSOFS）。这两种算法的最大区别在于 CMDPSOFS 在迭代过程中保持了粒子局部最优的继承性，更符合粒子群优化算法的思想，而不是通过排序使得每次迭代后的新粒子与前代几乎毫无关联。测试也表明 CMDPSOFS 在大多数情况下获得比 NSPSOFS 更优的解。然而 CMDPSOFS 中使用的均匀变异使算法的随机性大大增加，产生较多适应值差的解，降低了算法的收敛速度。

本文提出一种改进的 CMDPSOFS-II 算法，该算法将差分进化[13]算法作为变异算子引入到 CMDPSOFS 算法中，替换均匀变异生成新粒子，并加入差分进化的选择操作。该算法引入了差分进化的变异和选择方式，使得算法可以在迭代过程中基于向量差分的特点自适应选择变异的步长，保持了群体多样性，并提高了变异操作的效率。通过对一系列特征选择问题进行测试比较，实验结果表明这种改进的 CMDPSO-II 算法在特征选择上得到比原来的方法更优的结果，更好地平衡了全局和局部搜索能力。

1　背景

1.1　粒子群优化算法

粒子群优化算法最早由 Kennedy 等人[14]提出，算法初始化为一群随机粒子(随机解)，每个粒子拥有自身的速度和位置。之后对整个粒子群体进行迭代，在每次迭代中，每个粒子通过自身所找到的最优解 *pbest* 和全局最优解 *gbest* 来更新自己。设粒子 i 在维度 j 的速度为 $v_{i,j}$，位置为 $x_{i,j}$，粒子的速度更新公式和位置更新公式如式（1）（2）所示。

$$v_{i,j}=\omega v_{i,j}+c_1r_1\left(p_{i,j}-x_{i,j}\right)+c_2r_2\left(g_{i,j}-x_{i,j}\right) \quad (1)$$

$$x_{i,j}=x_{i,j}+v_{i,j} \quad (2)$$

其中：ω 为惯性权重，用来控制上次的进化结果对本次进化的影响程度；c_1 和 c_2 为加速系数，作用是衡量粒子的历史最优对其进化的引导程度；r_1 和 r_2 为[0,1]内的随机数；$p_{i,j}$ 和 $g_{i,j}$ 分别为粒子 i 在维度 j 的个体最优解 *pbest* 和全局最优解 *gbest*；$v_{i,j}$ 被一个预设的最大速度 $v_{\max}$ 所限制，$v_{ij,}\in[-v_{\max}, v_{\max}]$。当得出预设的结果或者达到预设的迭代次数时，算法结束。

1.2　多目标优化

当一个最优解的选择需要权衡两个或以上的目标时，而且这些目标之间存在相互矛盾的关系，那么这个问题称为多目标优化问题。多目标优化问题包含最大化或最小化问题，多目标最小化问题可以用式(3)表示，最大化问题与最小化问题类似，因此本文选取最小化问题进行研究。

$$\text{最小化}\quad F(x)=[f_1(x),f_2(x),\ldots,f_k(x)] \quad (3)$$

其中：x 为决策向量；$f_i(x)$为关于 x 的目标函数；k 为需要优化的目标数。

当决策向量 u 和 v 满足以下两个条件时，则称 u 支配 v。

$$\forall i: f_i(u)\le f_i(v) \quad i\in\{1,2,3,\ldots k\} \quad (4)$$

$$\exists j: f_j(u)\le f_j(v) \quad j\in\{1,2,3,\ldots k\} \quad (5)$$

如图 1 所示的双目标最小化问题，可知 x_1 既支配 x_2 又支配 x_3，但 x_2 与 x_3 互不支配。当一个解不受其他任何解支配时，称这个解为柏拉图最优解（Pareto-optimal solution）。所有柏拉图最优解在搜索空间组成的表面称为柏拉图前沿（Pareto front）。图 1 给出了两目标问题对应的柏拉图前沿曲线。图中的实心点代表柏拉图前沿上的解，它们相互之间是互不支配且不被其他解支配。

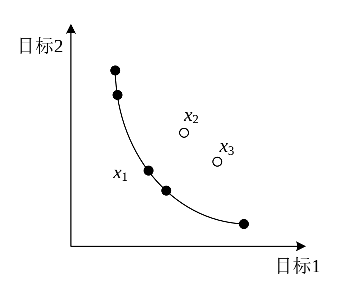


图 1　双目标最小化问题

Fig.1　Bi-objective minimization problem

1.3　差分进化

1996 年，Storn 和 Price 提出了差分进化算法（Differential evolution， DE）[15]，它是以随机多维数据为基础的优化算法。该算法主要思想是在群体内中通过个体差异变异产生待选个体，再经过交叉和选择操作以达到群体的进化。由于具有较好的全局收敛性和鲁棒性，非常适合用来求解最优化问题。

算法主要分为初始化、变异、交叉和选择步骤。首先在预设的空间内初始化群体的每个个体，设种群规模为 P，D 为优

化问题的维数。第 i 个个体的向量表示为 $x_i^G=(x_{i,1}^G,x_{i,2}^G,x_{i,3}^G,\ldots,x_{i,D}^G)$，其中 G 为个体 i 的代数。群体中每个个体的向量都通过式（6）进行变异操作。

$$v_i^{G+1}=v_{r1}^G+K(v_{r2}^G-v_{r3}^G) \tag{6}$$

其中：r_1、r_2、r_3 为[1,P]中互不相同且不同于 i 的随机数；K 称为变异因子，为[0,2]间的常量。利用式（7）中对变异后的个体 $v_{i,G+1}$ 进行交叉操作。

$$u_{i,j}^{G+1}=\begin{cases}v_{i,j}^{G+1} & \text{如果 } rand_{i,j}\le CR \text{ 或 } j=I_{rand}\\ x_{i,j}^G & \text{否则}\end{cases} \tag{7}$$

其中：$i=1,2,\ldots,P;\ j=i=1,2,\ldots,D$；$rand_{j,i}$ 为[0,1]间的随机数；CR 为[0,1]间的交叉概率；I_{rand} 在[1,2,...,D]内随机选择。$rand_{j,i}$ 的存在确保了交叉个体 $u_{i,G+1}$ 与父代个体 $x_{i,G}$ 不完全相同。然后利用式（8）进行选择操作，其中：f 为目标函数，这里的 f 为最小化问题。

$$x_i^{G+1}=\begin{cases}u_i^{G+1} & \text{如果 } f(u_i^{G+1})\le f(x_i^G)\\ x_i^G & \text{否则}\end{cases} \tag{8}$$

其中：$i=1,2,\ldots,P$。

差分进化算法简单且通用，具有利用个体局部信息和全局信息指引算法进一步搜索的能力，在许多情况下，都可以很容易地与其他算法进行混合从而生成性能更优的算法，如陈颖等人[16]提出的基于并行差分进化算法，差分进化算法同时也被研究用来选择支持向量机的参数[17]。

2　多目标优化粒子群特征选择算法

2.1　问题定义

特征选择是指从一个拥有一定特征数的数据集中选择一组最优特征的过程。但由于各特征之间的关系比较复杂，大量的特征之间的组合数量过多，无法将每个组合都进行评估，所以特征选择需要选用可行的算法进行优化。特征选择的困难在于特征之间的组合，一个有关的特征与其他特征组合到一起时，可能会变成多余的，或者无关的或者多余的特征与其他特征组合到一起时，可能会变成有关的。理想的特征子集应该是特征之间组合起来能够正确地分类不同的实例的互补特征集。

目前，虽然许多针对特征选择的算法被提出，大部分算法的目标只是降低分类错误率。特征数量的增加虽然可以降低分类错误率，但是过多的特征数将导致分类变得困难，也不利于后续的数据分析。如何在提高可接受的分类错误率的同时，降低特征数量，这是两个冲突的目标。多目标优化技术利用算法对这两个目标同时优化，可以得到一组多目标最优解，也就是柏拉图前沿解。通过对这些解的进一步分析，可以折中选择特征数量较少，但分类错误率相对较低的特征组合，用于实际的分类应用中。

本文研究的多目标特征选择问题以分类错误率 f_1 和特征数量 f_2 作为两个分类优化目标，定义如下：

$$\text{最小化}\quad F(x)=[f_1(x),f_2(x)] \tag{9}$$

其中：

$$f_1=\frac{FP+FN}{TP+TN+FP+FN} \tag{10}$$

$$f_2=|V| \tag{11}$$

式（10）中的 P 或 N 代表观察样本（真实样本）属于或不属于某个类别，T 或 F 代表预测结果。如果该样本的观察与预测结果一致就为 T，否则为 F。该式子的计算值就是分类错误率。式（11）对应的是被选中的特征集合 V 中元素的个数。

针对上述双目标特征选择优化问题，文献[12]提出了基于非支配排序的特征选择粒子群（NSPSOFS）算法和基于拥挤、变异和支配的特征选择粒子群算法（CMDPSOFS）算法。下面先介绍这两种算法的实现方式，第 3 章将描述本文提出的改进算法的实现。

2.2　NSPSOFS 算法

NSPSOFS 算法将基于非支配排序的多目标粒子群算法应用到特征选择上，其中两个最重要的步骤是迭代中对全局最优解（*gbest*）的选择和对群体的更新。在每次迭代中，NSPSOFS 首先评估群体中每个粒子的适应值（特征数和分类错误率），再根据适应值确定群体中的非支配粒子集。计算每个非支配粒子的拥挤距离，然后将非支配粒子按拥挤距离降序进行排序。更新群体中的粒子时，在拥挤距离最小的非支配粒子中随机选择出 *gbest*，而个体最优解（*pbest*）则是粒子在每次迭代中都不被当前粒子所支配的解。当粒子更新后的解支配自身的 *pbest* 时，则替换 *pbest*。

粒子确定了 *pbest* 和 *gbest* 后，根据式（1）（2）更新自身的位置和速度。更新后的粒子与更新前的粒子都被添加到一个集合 *union* 中，然后在 *union* 中将所有粒子根据不同的非支配级别分到子集 $F=(F_1,F_2,F_3,\ldots,F_k)$中，$k$ 代表最多的非支配子集数量。清空群体，从子集 F_1 开始将其中的粒子添加到群体中。若群体所需的粒子数大于当前的非支配子集粒子数，则将当前非支配子集都添加到群体中；否则，将当前非支配子集的粒子按拥挤距离降序排序并添加到群体中，直到群体的粒子数达到预设的群体规模 P。

2.3　CMDPSOFS 算法

由于 NSPSOFS 更新群体的方式存在导致粒子多样性减小的缺陷，而且每次迭代都对粒子排序会导致粒子记录的 pbest 并非通过自身的解产生，Xue 等人[12]进一步研究了添加基于拥挤、变异和支配方法的多目标粒子群优化算法 CMDPSOFS。该算法更加符合粒子群优化算法是思想，在迭代过程中保持了粒子局部最优的继承性。

图 2 给出了 CMDPSOFS 算法流程。其中 N 代表群体规模。该算法使用一个领导集合（*LeaderSet*）来保存非支配解，每个粒子的 *gbest* 在 *LeaderSet* 中用二元竞赛方式在拥挤距离最小的粒子中选择。在每一代群体中的非支配粒子对 *LeaderSet* 进行

更新并加入到档案集合 *Archive* 中。算法迭代执行，直到满足停止条件。

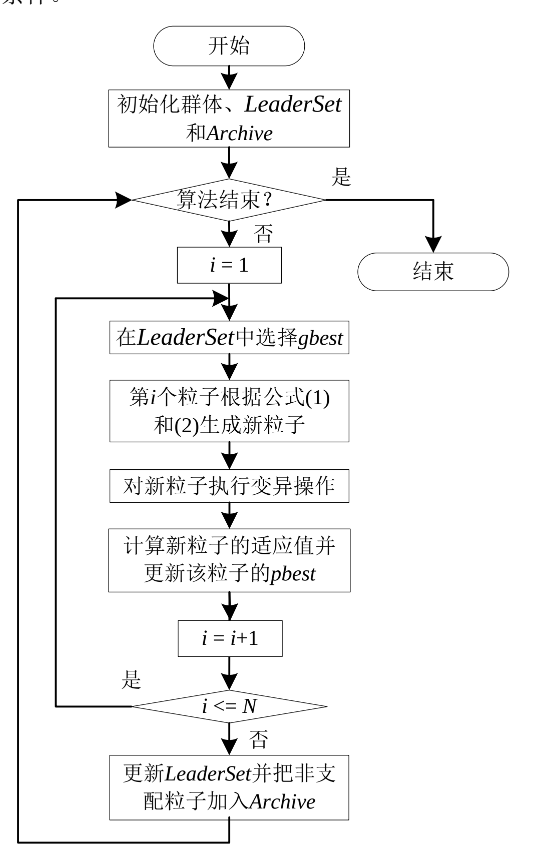


图 2　CMDPSOFS 算法流程

Fig.2　Flowchart of CMDPSOFS

CMDPSOFS 采用了变异操作增加群体多样性。在变异步骤中，CMDPSOFS 将群体的粒子随机等分为三组：

a）第一组不做任何变异；

b）第二组采用均匀变异方式，总特征数的倒数作为变异概率，在向量的定义域内随机取值得到变异向量；

c）第三组为非均匀变异，变异概率与第二组相同，通过随着迭代次数的增加，缩小变异选值范围，每次在范围内随机得到一个变异值，这种变异方式由于其选值范围会随时间变化而变小，到后期阶段则非常局部化。

3　基于差分进化改进的 CMDPSOFS-II 算法

3.1　对 CMDPSOFS 的改进

本文提出的 CMDPSOFS-II 算法将 CMDPSO 算法与差分进化算法中是变异和选择方式相结合，应用到特征选择中。在 CMDPSOFS 算法的变异步骤的基础上，将均匀变异算子替换为差分进化算法的变异算子并添加了差分进化的选择步骤，具体流程如图 3 所示。

CMDPSOFS-II 使用差分变异算子对粒子进行变异时，如果出现变异后的值超过变量的预设范围[l_k, u_k]时，则有 50%的概率取预设范围的中间值，即(l_k+u_k)/2，或者 50%的概率取变异前的值。由于随机选择父代个体进行差异重组，算法得到的变异粒子只使用了父群体中的两个随机粒子差异向量进行修正，保持了算法的多样性。

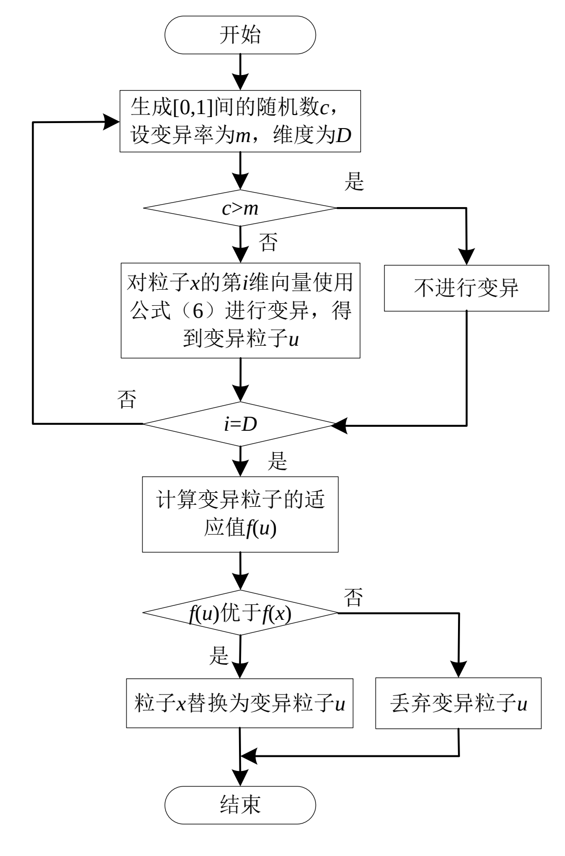


图 3　差分变异流程

Fig.3　Flowchart of differential mutation

在 CMDPSOFS-II 的差分变异中还增加了选择操作，当变异后的适应值优于父代适应值时，选择变异后的粒子，否则选择父代粒子，此操作加快了算法的收敛速度。

3.2　CMDPSOFS-II 算法流程

步骤 1）初始化群体、*LeaderSet* 和 *Archive*，计算 *LeaderSet* 中粒子的拥挤距离，随机将群体中的粒子等分为三组。

步骤 2）按预设的迭代次数进行循环

2.1）对群体中每个粒子：

i.　在 *LeaderSet* 中使用二元竞赛根据拥挤距离选择粒子的 *gbest*

ii.　更新粒子的速度与位置

iii.　执行变异操作

1）属于第一组的粒子不进行操作

2）属于第二组的粒子进行本文提出的差分变异，评估变异后粒子的适应值，若变异后的粒子支配当前粒子，则将变异后的粒子替换为当前粒子

3）属于第三组的粒子进行非均匀变异

iv.　计算每个粒子的适应值，更新粒子的 *pbest*

2.2）确定群体中的非支配粒子，并用来更新 *LeaderSet*

2.3）将 *LeaderSet* 中的粒子保存到 *Archive*

2.4）计算 *LeaderSet* 中每个粒子的拥挤距离

步骤 3）计算 *Archive* 中每个粒子的适应值（特征数和分类

错误率）并作为结果返回

算法中的惯性权重 ω 和加速系数 c_1 和 c_2 与 CMDPSOFS 中的设置相同，即变异率为 $1/n$，n 为优化问题维数，也就是特征数。差分变异中的变异因子 K 取 0.5。

4　实验设计及数据分析

4.1　数据集与参数设置

本文在 UCI machine learning repository[18]中选取了表 1 所示的数据集，每个数据集分为训练集和测试集，分别为总实例数的 70%和 30%。实验比较了本文提出的 CMDPSOFS-II 和 NSPSOFS[12]、CMDPSOFS[12]。在测试中，所有算法都采用支持向量机来对数据进行分类训练和测试，软件采用的是 Chih-Chung 研发的 LIBSVM[19]，运行平台为 Java。

表 1　数据集

Table 1　Data set

数据集	总特征数	类别数	实例数
Wine	13	3	178
Australian	14	2	690
Zoo	17	7	101
Vehicle	18	4	846
German	24	2	1000
WBCD	30	2	569
Ionosphere	34	2	351
Hill-valley	100	2	606

4.2　实验结果

根据文献[12]，在参与测试的算法中，v_{max}=0.6，种群大小 N=30，最大迭代次数 T=500，用来决定特征是否被选中的 θ=0.6。在 NSPSOFS 中，惯性权重 ω=0.7298，加速系数 c_1=c_2= 1.49618。在 CMDPSOFS 和 CMDPSOFS-II 中，惯性权重 ω 为[0.1,0.5]间的随机数，加速系数 c_1 和 c_2 为[1.5,2.0]间的随机数，变异率设置为 $1/n$。所有的算法都在测试中独立运行 30 次。图 4 比较了三种算法最优的非支配解集对应的柏拉图前沿。图 4 中的曲线分别表示 CMDPSOFS、NSPSOFS 和 CMDPSOFS-II 对相应数据集独立运行 30 次得到的最优非支配结果。图中的子标题表示数据集名称，括号内的数据表示总特征数和使用数据集全部特征时的分类错误率。使用全部特征可能会被无关的特征影响训练的准确性，因此可以看到算法优化得到的解的分类错误率有可能比使用全部特征得到的分类错误率要低。

首先分析 NSPSOFS 的测试结果，除了 Ionosphere，在其余的数据集中都能得到分类错误率优于使用全部特征时的分类错误率，且特征数要小于总特征数。特别地，在 WBCD 数据集中只需要选取 10%的特征数就得达到比使用全部特征得到更小的分类错误率，即 30 个特征种选择了 3 个，在 Vehicle 中只选取了 11%的特征，即 18 个特征种选择了 2 个，而且错误率低于总分类错误率。在测试的数据集中，NSPSOFS 平均能将特征数减小到总特征数的 20%。

CMDPSOFS 在每个数据集的测试中都能至少得到一个分类错误率小于使用全部特征时的分类错误率的特征子集。CMDPSOFS 使特征集的平均特征数降低到总特征数的 22%。其中在数据集 WBCD 和 Australian 中，CMDPSOFS 都选择了总特征数的 7%，即在 WBCD 中 30 个特征选择了 2 个，在 Australian 中 14 个特征选择了 1 个，同时分类错误率低于使用全部特征时的分类错误率。在大多数解集中，相对于 NSPSOFS 算法，CMDPSOFS 得到的非支配解都比较多且分布均匀，而且在相同特征数下，CMDPSOFS 得到的特征得到的分类错误率要比 NSPSOFS 要低，也就是说 CMDPSOFS 得到的特征要比 NSPSOFS 得到的特征要好。

在 CMDPSOFS-II 的实验结果中，在每个数据集的测试中都能至少得到一个分类错误率小于使用全部特征时的分类错误率的特征子集。CMDPSOFS-II 平均使特征集的特征数降低到总特征数的 14%。在数据集 Vehicle 中选择了总特征数的 6%（1/18），即只选择了 1 个特征，在 WBCD 中选择了总特征数的 7%（2/30），同时分类错误率低于使用全部特征时的分类错误率。

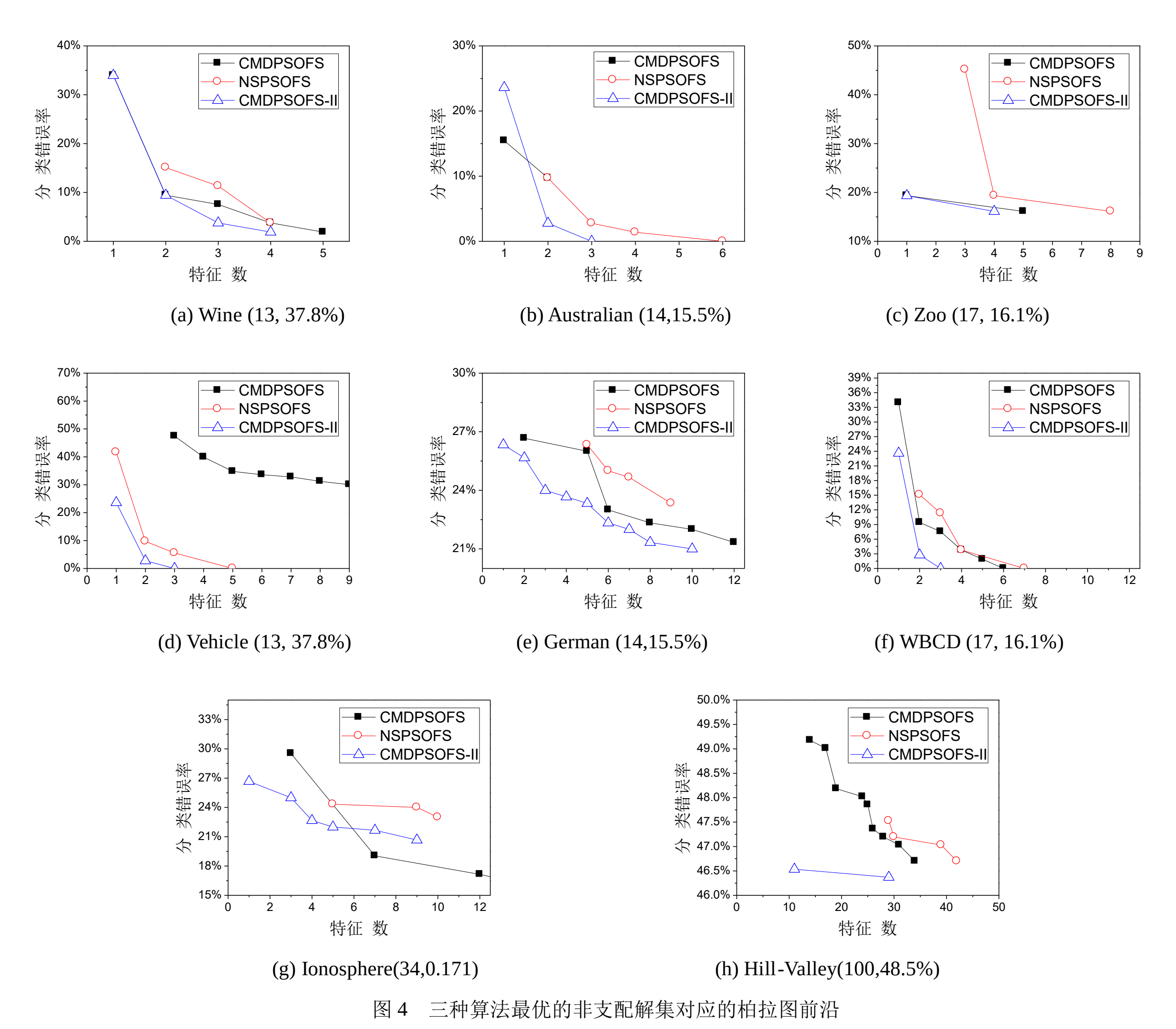
表 2 列出了三种算法在测试结果中错误率低于使用所有特征的最小特征数，括号内为分类错误率。可以看出，CMDPSOFS-II 在多数情况下能得出更低的特征数，在特征数相等的时候，CMDPSOFS-II 可以选出错误率更低的特征组合，特别在 Hill-Valley 中，CMDPSOFS-II 得出的特征数和错误率均低于 NSPSOFS 和 CMDPSOFS。

对比 CMDPSOFS-II、NSPSOFS 和 CMDPSOFS，CMDPSOFS 在解的多样性、特征数和分类错误率的优化上对比 NSPSOFS 都有明显的优势。在所有的情况下，CMDPSOFS-II 在特征数和分类性能的优化上都优于 NSPSOFS，例如在 German 数据集中，当特征数相同时，CMDPSOFS-II 得出的分类错误率比 NSPSOFS 低。除了在 Ionosphere 和 Australian，CMDPSOFS 在少数解上优于 CMDPSOFS-II 之外，在其他测试中 CMDPSOFS-II 得出的非支配解集都优于 CMDPSOFS。CMDPSOFS-II 不仅继承了 CMDPSOFS 的群体多样性，还提高了特征数和分类性能的优化能力。

表 2　错误率低于使用所有特征的最小特征数及对应的分类错误率

Table 2　Minimum number of features with error rate lower than using all features and corresponding classification error rate

	NSPSOFS	CMDPSOFS	CMSPSOFS-II
Wine(13,37.8%)	2 (15.1%)	2 (9.4%)	1 (33.9%)
Australian(14,15.5%)	2 (9.7%)	1 (15.5%)	2 (2.8%)
Zoo(17,16.1%)	8 (16.1%)	5 (16.1%)	4 (16.1%)
Vehicle(18,28.9%)	3 (5.6%)	10 (28.1%)	3 (0%)
German(24,25.7%)	6 (25.0%)	6 (23.0%)	2 (25.6%)
WBCD(30,14.0%)	3 (11.3%)	2 (9.4%)	2 (2.8%)
Ionosphere(34,17.1%)	NA	12 (17.1%)	NA
Hill-Valley(100,48.5%)	29 (47.5%)	19 (48.2%)	11 (46.5%)



(a) Wine (13, 37.8%)　(b) Australian (14,15.5%)　(c) Zoo (17, 16.1%)

(d) Vehicle (13, 37.8%)　(e) German (14,15.5%)　(f) WBCD (17, 16.1%)

(g) Ionosphere(34,0.171)　(h) Hill-Valley(100,48.5%)

图 4　三种算法最优的非支配解集对应的柏拉图前沿

Fig.4　Pareto fronts of optimal non-dominance sets of three algorithms

5　结束语

CMDPSOFS-II 是基于差分进化操作，针对应用在特征选择的 CMDPSOFS 算法提出的优化算法。由于在群体的中一定部分粒子中引入差分进化操作中的变异和选择机制，使 CMDPSOFS-II 继承了 CMDPSOFS 的粒子多样性，且能在一定程度上抑制算法随机性，提高了变异的有效性，加快了 CMDPSOFS 的收敛速度。

参考文献：

[1] Dash B M, Liu H. Feature selection for classification [J]. Intelligent Data Analysis, 1997, 1 (3): 131-156.

[2] Unler A, Murat A. A discrete particle swarm optimization method for feature selection in binary classification problems [J]. European Physical Journal Applied Physics, 2009, 206 (3): 528-539.

[3] Fong S, Wong R, Vasilakos A V. Accelerated PSO swarm search feature selection for data stream mining big data [J]. IEEE Trans on Services Computing, 2016, 9 (1): 33-45.

[4] Naeini A A, Babadi M, Mirzadeh S M J, *et al.* Particle swarm optimization for object-based feature selection of VHSR satellite images [J]. IEEE Geoscience and Remote Sensing Letters, 2018, 15 (3): 379-383.

[5] Han Fei, Yang Chun, Wu Yaqi, *et al.* A gene selection method for microarray data based on binary PSO encoding gene-to-class sensitivity information [J]. IEEE//ACM Trans on Computational Biology and Bioinformatics, 2017, 14 (1): 85-96.

[6] Wang Hong, Niu Ben. A novel bacterial algorithm with randomness control for feature selection in classification [J]. Neurocomputing, 2017, 228: 176-186.

[7] Belciug S, Serbanescu M S. Regression-based approach for feature selection in classification issues. application to breast cancer detection and recurrence [J]. ACTA Universitatis Cibiniensis Technical Series, 2015, 67 (1): 13-18.

[8] 段洁，胡清华，张灵均，等. 基于邻域粗糙集的多标记分类特征选择算法 [J]. 计算机研究与发展, 2015, 52 (1): 56-65. (Duan Jie, Hu Qinghua, Zhang Lingjun, *et al.* Feature selection for multi-label classification based on neighborhood rough sets [J]. Journal of Computer Research and Development, 2015, 52 (1): 56-65.)

[9] 谢娟英，谢维信. 基于特征子集区分度与支持向量机的特征选择算法



[J]. 计算机学报, 2014, 37 (8): 1704-1718. (Xie Juanying, Xie Weixin. Several feature selection algorithms based on the discernibility of a feature subset and support vector machines [J]. Chinese Journal of Computers, 2014, 37 (8): 1704-1718.)

[10] Zhang Tao, Ding Biyun, Zhao Xin, *et al.* A fast feature selection algorithm based on swarm intelligence in acoustic defect detection [J]. IEEE Access, 2018, 6: 28848-28858.

[11] Zhang Yong, Gong Dunwei, Cheng Jian. Multi-objective particle swarm optimization approach for cost-based feature selection in classification [J]. IEEE//ACM Trans on Computational Biology and Bioinformatics, 2017, 14 (1): 64-75.

[12] Xue Bing, Zhang Mengjie, Browne W N. Particle swarm optimization for feature selection in classification: a multi-objective approach [J]. IEEE Trans on Cybernetics, 2013, 43 (6): 1656-1671.

[13] Storn R, Price K. Differential evolution: a simple and efficient adaptive scheme for global optimization over continuous spaces [R]. Berkeley: University of California, 2006.

[14] Kennedy J, Eberhart R. Particle swarm optimization [C]// Proc of IEEE Neural Networks. 1995: 1942-1948.

[15] Storn R. On the usage of differential evolution for function optimization [C]// Proc of Fuzzy Information Processing Society. 1996: 519-523.

[16] 陈颖，林盈，胡晓敏. 多种群多策略的并行差分进化算法 [J]. 计算机科学与探索, 2014, 8 (12): 1502-1510. (Chen Ying, Lin Ying, Hu Xiaomin. Parallel differential evolution with multi-population and multi-strategy [J]. Journal of Frontiers of Computer Science & Technology, 2014, 8 (12): 1502-1510.)

[17] 陈涛，雍龙泉，邓方安，等. 基于差分进化算法的支持向量机参数选择 [J]. 计算机工程与应用, 2011, 47 (5): 24-26. (Chen Tao, Yong Longquan, Deng Fang’an, *et al.* Parameters selection of support vector machine based on differential evolution [J]. Computer Engineering and Applications, 2011, 47 (5): 24-26.)

[18] Lichman, M. UCI Machine Learning Repository [DB/OL]. (2007) [2013-04-04]. http://archive. ics. uci. edu/ml.

[19] Chang Chih Chung, Lin Chih Jen. LIBSVM: a library for support vector machines [EB/OL]. (2011) [2016-12-22]. http://www. csie. ntu. edu. tw/~cjlin/libsvm.